TIBCO® Patterns

User's Manual

Software Release 4.5.1
September 2011

Important Information

All other product and company names and marks mentioned in this document are the property of their respective owners and are mentioned for identification purposes only.

THIS SOFTWARE MAY BE AVAILABLE ON MULTIPLE OPERATING SYSTEMS. HOWEVER, NOT ALL OPERATING SYSTEM PLATFORMS FOR A SPECIFIC SOFTWARE VERSION ARE RELEASED AT THE SAME TIME. SEE THE README.TXT FILE FOR THE AVAILABILITY OF THIS SOFTWARE VERSION ON A SPECIFIC OPERATING SYSTEM PLATFORM.

THIS DOCUMENT IS PROVIDED .AS IS. WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT.

THIS DOCUMENT COULD INCLUDE TECHNICAL INACCURACIES OR TYPOGRAPHICAL ERRORS. CHANGES ARE PERIODICALLY ADDED TO THE INFORMATION HEREIN; THESE CHANGES WILL BE INCORPORATED IN NEW EDITIONS OF THIS DOCUMENT. TIBCO SOFTWARE INC. MAY MAKE IMPROVEMENTS AND/OR CHANGES IN THE PRODUCT(S) AND/OR THE PROGRAM(S) DESCRIBED IN THIS DOCUMENT AT ANY TIME.

THE CONTENTS OF THIS DOCUMENT MAY BE MODIFIED AND/OR QUALIFIED, DIRECTLY OR INDIRECTLY, BY OTHER DOCUMENTATION WHICH ACCOMPANIES THIS SOFTWARE, INCLUDING BUT NOT LIMITED TO ANY RELEASE NOTES AND "READ ME" FILES.

This Product is covered by U.S. Patent No. *put number here*, *put number here*, and *put number here*.

Contents

1 Introduction

This document explains how to install and run the TIBCO PATTERNS family of products. The product names, the associated executable names and a description of the members of this family are given below. In the executable name "*n.n.n*" is replaced with the release number and "*platform*" is replaced with a platform label indicating the machine and OS environment the executable is intended to run on:

Label	*Environment*
aix32	AIX 5.1 through AIX 5.3 POWER PC Systems. 32 Bit Executable.
aix64	AIX 5.1 through AIX 5.3 POWER PC Systems. 64 Bit Executable.
hpux32	HPUX 11 series Itanium Systems. 32 Bit Executable.
hpux64	HPUX 11 series Itanium Systems. 64 Bit Executable.
linux32	Linux on x86/AMD processors. 32 Bit Executable.
linux64	Linux on x86/AMD processors. 64 Bit Executable.
solaris32	Solaris 10 SPARC systems. 32 Bit Executable.
solaris64	Solaris 10 SPARC systems. 64 Bit Executable.
win32.exe	Microsoft Windows® XP/Vista/System 7 x86/AMD Systems. For 32 Bit systems.
win64.exe	Microsoft Windows® XP/Vista/System 7 x86/AMD Systems. For 64 Bit systems.

Product Name	*Executable Name and Description*
TIBCO® PATTERNS - SEARCH	`TIB_patterns_search_`*n.n.n_platform* Matching Engine Only.
TIBCO® PATTERNS - SEARCH + LEARN	`TIB_patterns_searchlearn_`*n.n.n_platform* Matching Engine and Machine Learning Scoring.
TIBCO® PATTERNS - LEARN	`TIB_patterns_learn_`*n.n.n_platform* Machine Learning Scoring Only. No Matching.

The term TIBCO PATTERNS ENGINES is used to indicate any one of the products in the family. The term TIBCO PATTERNS EXECUTABLE is used to indicate any of the executables.

2 Installation

Installation is trivial. All that is needed to run the product is the executable file. So all that needs to be done to install the product is to copy the executable file to whichever directory it is to be executed from.

3 Running the Server

Here is the command syntax for the server:

```
TIB_patterns [-p port] [-t maxthreads] [-R restoredir] [-A] [-a common-dir]
           [-c console] [-l querylog] [-r pidfile]
           [-v] [-d] [-S] [-s] [-D socketlog] [-F gip|sort]
           [-m memory-cap] [-L max-loads] [-P B|I|O|N] [-T sub-limit]
           [-M model-file] [-h] [-?] [ipaddr ...]
```

Options and arguments:

`-p` port — Listen on the specified port. The default port is 5051.

`-t` maxthreads — Spawn no more than the specified number of threads. The default is 4.

`-R` restoredir — Enable checkpoint/restore using the indicated directory as the checkpoint/restore directory. The indicated directory must already exist. If this argument is not given the checkpoint feature is disabled.

`-A` — Auto restore. Upon start up restore all checkpointed tables in the restore directory.

`-a` common-dir — Restore from a common directory. Upon start up all checkpointed tables in the indicated directory are restored. Checkpointing is *not* enabled by this option, although it may be used with the `-R` and `-A` options.

`-c` console — Write error messages to the specified file. By default, the server sends messages to the syslog on Unix and the eventlog on Microsoft Windows®.

`-l` logfile — Log incoming queries to the specified file.

`-r` pidfile — Write the server's process ID to the specified file. The file is deleted when the server terminates. On Microsoft Windows® platforms the file is created and cleaned up but contains a dummy value.

`-v` — Verbose mode. Write a notice to the console log for every connection request received.

`-d` — Start for debugging. The server *doesn't* detach itself from the controlling terminal, nor does it put itself into the background.

`-S` — *Windows only:* Run as a console application.

`-s` — Single threaded mode. Do not use multiple threads. The same as `-t 0`.

`-D` socketlog — Log all messages sent and received by the server to the indicated file. Use only at the request of your TIBCO representative. This will produce a massive file.

`-F` gip *or* sort — Select the prefilter to use. This is one of "`gip`" or "`sort`". The default for the TIBCO® Patterns - Search is "`gip`". The default for the TIBCO® Patterns - Search + Learn is "`sort`". This argument is not valid for the TIBCO® Patterns - Learn engine.

`-m` memory-cap — Set a limit for the total amount of space used for all data base tables in the engine. The space is given in kilobytes. The minimum allowed size is 1024. When the total memory used for all data base tables in the process has reached the limit specified any attempt to add additional records or tables will be disallowed. If this argument is not given no check is made, if enough memory can not be allocated for an additional record or table the process will terminate. Thus you can use this argument to protect the server from aborting due to insufficient memory resources or to cap its memory resource use on shared systems.

`-L` max-loads Set the limit for the number of parallel loads in a multi-table restore. By default the Auto restore on start up will load up to *maxthreads/3* tables in parallel to reduce load time. During command processing parallel loads of multi-table restores are turned off by default as they could interfere with other command processing. This can be controlled through this option. Setting this to a positive value will allow up to that many tables to be loaded in parallel during command processing. Setting it to zero removes all restrictions, all tables are loaded in parallel. Setting it to a negative value disallows parallel loading in multi-table restores during command processing (same as default behavior). For the initial Auto-load the limit is always the *maximum* of *maxthreads/3* or *max-loads.*

`-P` socket-option Use persistent socket connections. socket-option is one of I = Incoming only, O = outgoing only, B (default) = both, N = neither.

`-T` sub-limit This specifies a limit on the number different words or phrases in a query string to which thesaurus substitutions will be applied. Substitutions beyond the limit are simply ignored. There can be a large performance penalty for very large numbers of substitutions, this limits that penalty. The default is 40, which should be far more than enough for most applications.

`-M` modmname Many TIBCO® Patterns - Learn Models come with metadata to identify when the model was created, who created it, its intended purpose. Use this command line option to print out the metadata, and then exit.

`-h` or `-?` Help. Print a usage message and exit.

ipaddr ... Accept connections from the specified IP addresses as well as the server host. By default, the server accepts connections only from `localhost` (127.0.0.1).

The `-M` option is valid only on the TIBCO® Patterns - Search + Learn and TIBCO® Patterns - Learn servers. The `-R`, `-A`, `-a`, `-l`, `-m`, `-T`, `-L`, and `-F` options are not valid on the TIBCO® Patterns - Learn server.

Unix only: On startup, the server detaches itself from the controlling terminal and redirects all output to the console and logfiles as specified in the command line arguments. This is the standard behavior for background "demon" processes on UNIX platforms. The `-d` option disables this behavior.

The list of IP addresses consisting of `localhost` (127.0.0.1) plus any addresses given on the server command line is called the *authentication list.* Only hosts in the authentication list may connect to the server. Addresses may include wild cards and subnet mask lengths: `129.48.34.* 192.168.*.* 129.48.34.0/24`.

4 Checkpoint/Restore Directories

To use the Checkpoint/Restore feature a unique checkpoint/restore directory must be allocated for each running TIBCO Patterns Engines. This directory is where the data on the checkpointed tables is stored. No two running TIBCO Patterns Engines may share the same checkpoint/restore directory. Nor should any other process attempt to modify this directory or its contents in anyway. The first time the directory is used it must be empty (have no files). After that management is under control of the TIBCO Patterns Engines and performed automatically, no additional management is needed.

In a typical production environment there would be one TIBCO Patterns Engines per machine. A checkpoint/restore directory would be created on the machine. The server is then started using the "`-R` *directory*" argument, where *directory* is the full path name to the created directory. The TIBCO Patterns Engines then takes over full responsibility for managing the contents of the directory, no additional management should be needed. When a server starts up it will perform any necessary clean up of the directory[1].

4.1 Restoring from a common directory

As noted above each running TIBCO Patterns Engines must have its own unique checkpoint/restore directory. Sometimes if you are running multiple TIBCO Patterns Engines containing the same tables[2] it may be convenient to keep a single copy of the tables and have all TIBCO Patterns Engines load from the common copy. This can be done by using the `-a` option. This loads all of the checkpointed tables from the named directory *but does not enable checkpointing in the directory.* It is treated as a read only directory. The TIBCO Patterns Engines will validate the contents of the directory, *but will not attempt to clean it.* Instead the TIBCO Patterns Engines exits if the directory contents indicate one or more checkpoints may be incomplete.

You can both restore from a common directory *and* enable checkpoint restore by using both the `-R` and `-a` options. You can even use the auto restore option (`-A`) so that tables from both the common directory and the checkpoint/restore directory will be loaded on start up. The common tables are loaded first, followed by the checkpoint/restore tables, which may overwrite the common tables.

One way to manage a set of multiple TIBCO Patterns Engines with a common set of tables would be to have one TIBCO Patterns Engines assigned as the master. This engine would always be started first and would have checkpoint/restore enabled on the directory (use the `-R` and `-A` options). When it starts it would ensure the directory is cleaned up. Once the primary engine has cleaned the directory (usually a matter of seconds) the other TIBCO Patterns Engines can be started using the `-a` option to load the tables. *Note however that if the master* TIBCO Patterns Engines *performs checkpoint operations while another engine is restoring tables it could cause the other engine to fail and exit.* This is especially true if a table is deleted.

4.2 Selecting a Checkpoint/Restore Directory

Things to consider when allocating a checkpoint/restore directory:

Accessibility. The TIBCO Patterns Engines must be able to access the directory from the home directory where the server was started.

File Name Limits. The file system should support long file names. The file names in the directory

[1] Provided no other process has modified the directory contents by adding, removing or modifying any of the files in the directory.

[2] This is often done to handle very large query loads.

will contain the table name plus 6 characters.

Space. The file system or partition on which the directory resides must have enough space to store the data for all of the tables that will be checkpointed. You should assume that the checkpoint files created in the checkpoint/restore directory will be about twice the size of the raw table data. So if a CSV file dump of the table data takes about 1Gig the checkpoint file may take up to 2Gig. The TIBCO Patterns Engines do *not* check for availability of space before writing out data, so it is critical that enough space be available on the device.

Size Restrictions. The data for a table is stored in a single file. For large tables this could exceed the maximum file size on some file systems. You should ensure that the file system supports large file sizes.

Network Traffic. If the directory is on a remote mounted file system (e.g. NFS) all of the data reads and writes involved in dumping and reading the checkpoint files could generate a very large amount of network traffic which could effect the overall network performance.

Speed. Directories on remote devices or slow devices could significantly impact checkpoint/restore performance. A fast local device, such as a good SATA disk drive, is best.

Reliability. You should not depend on the TIBCO Patterns Engines, even when using checkpoints, as your primary, secure, data repository. But if losing the checkpoint data could significantly impact the availability of critical services you may want to consider putting the checkpoint directory on a mirrored or RAID device.

Security. The data stored in the checkpoint files is *not* encrypted. Therefore the checkpoint/restore directory should not be accessible to anyone who is not authorized to see the data.

4.3 Checkpoint/Restore directory contents

The contents of the directory is managed by the TIBCO Patterns Engines, so generally there is no need to look in the directory. But as an FYI the files you may see in the directory are briefly described below.

file name	*description*
LockFlag	Used to coordinate access to the directory
table-name`-0.ntc`	Checkpoint file for table *table-name*. This is a binary data file and contains the complete information on the table, both the table structure and options and the table data.
table-name`-1.ntc`	A previous checkpoint file for table *table-name*. This should exist only if there was a crash of the system during a checkpoint operation. The TIBCO Patterns Engines will clean this up on start up.
table-name`-n.ntc`	An incomplete checkpoint file for the table *table-name*. This should exist only if there was a crash of the system during a checkpoint operation. The TIBCO Patterns Engines will clean this up on start up.

5 Server Threads

TIBCO Patterns Engines are multi-threaded. If your host machine has multiple processors, a TIBCO Patterns Engines process will use them in parallel. Even on a single-processor host, it can do database loads and dumps in parallel with queries.

How many threads TIBCO Patterns Engines use depends on the platform. To avoid resource contention, the number of request handler threads is limited to 4 by default. You can adjust this limit with the `-t` option on the command line. The optimal number of threads depends on how many CPUs are available, the hardware and memory configuration, the volume and configuration of data loaded into the server, the expected query rate, and the type of queries.

If you set the thread count to zero, the server handles all requests in the main thread. It may use separate threads for other processes like logging, but only one thread will read and process commands. A server that is being used as a gateway for multi-table searches should *never* be started with a single thread.

6 IPC Timeouts

The timeout for interprocess reads and writes for TIBCO Patterns Engines is fixed at 30 seconds. If a read or write to an open socket to the server is inactive for 30 seconds, the server closes the socket. Additionally, if a persistent connection is not used for 90 seconds, the server closes the socket[3].

7 Shutting Down the Server

The TIBCO Patterns Engines can be shut down in two ways:

- An authorized client may connect and issue the `svrshutdown` command.
- *Unix only:* You may send the signal `SIGTERM` to the engine. The Unix `kill` command sends `SIGTERM` by default.

In either case the engine will first stop accepting new commands, then wait for all currently running commands to complete, free all data, and then exit. Long running commands check for a termination request and exit early but it is possible in rare cases that the shutdown process could take a considerable amount of time.

To force a rapid shutdown of the engine send a second `SIGTERM` signal to the engine[4]. This will force the engine to do an immediate, unclean, shutdown. This is to be avoided as it may not properly clean up all client socket connections, potentially causing problems in the client processes, and won't clean up any PID file (see `-r` option).

[3] Note that the Java and .Net API's will time out an unused persistent connection in 30 seconds.

[4] You can't send another `svrshutdown` command as the server has stopped accepting commands.

8 Running TIBCO Patterns Engines on Microsoft Windows®

8.1 Running TIBCO Patterns Engines as a Microsoft Windows® Service

To run TIBCO Patterns Engines as a service, use the `sc` program to install the service. The command should look like this:

```
sc create "TIBCO Patterns Engine" binpath= c:\path_to_engine\TIB_patterns_search_4.5.1_win64.exe
```

Note the space after `binpath=`. Use the correct name for your TIBCO Patterns Executable and make sure to specify the correct path to TIBCO Patterns Executable. To remove the service, type:

```
sc delete "TIBCO Patterns Engine"
```

You can now use the services applet to control the service called "TIBCO Patterns Engine." The applet is in the Control Panel under **Administrative Tools.** With it you can start and stop the service, set it to start automatically on reboot, and specify command-line options.

Also in the Control Panel under **Administrative Tools** you find the Event Log applet. The TIBCO Patterns Engines logs start, stop and error messages in the Event Log. Use the applet to view them.

You can use the `-c` option to write the error messages to a file instead of the Event Log. Command-line errors will still appear in the Event Log.

You cannot use the `-M`, `-?` or `-h` options when you run the TIBCO Patterns Engines as a service.

By default the TIBCO Patterns Engines on Microsoft Windows® run as a service. To run the TIBCO Patterns Engines as a console application, use the `-S` option.

8.2 Microsoft Windows® Configuration for the TIBCO Patterns Engines

The default configuration for Microsoft Windows® Server 2000, Microsoft Windows® Server 2003 and Microsoft Windows® XP may not be appropriate for running the TIBCO Patterns Engines under heavy loads. Depending on the particular characteristics of your machine and usage some or all of the following configuration changes may be necessary to avoid intermittent network errors. *Note: These configuration changes don't apply to Microsoft Windows® Vista or Microsoft Windows® System 7.*

With the introduction of *persistent sockets* in release 4.2.3 of the TIBCO Patterns Engines the need for these configuration changes should be greatly reduced. Persistent sockets avoids, where possible, the creation of a new connection for each command. To make proper use of the *persistent socket* feature the client application must be long lived and reuse the same `NetricsServerInterface` object if using the Java or .Net interfaces.

8.2.1 Increase Ports Available:

DESCRIPTION:

The default settings on most Microsoft Windows® machines allocate a fairly small set of ports for applications making outbound TCP/IP connections (about 4,000). By default Microsoft Windows® will wait about 5 minutes after a port is closed before reusing it. As each request to the TIBCO Patterns Engines may make a new connection clients running on Microsoft Windows® can quickly exhaust the outgoing pool of ports. If this happens connection attempts will fail. These registry changes increase the number of ports available for outgoing connections to the maximum value supported by Microsoft Windows® and decreases the recycle wait time to the minimum value supported by Microsoft Windows®. This greatly decreases the chance of exhausting the pool of outgoing ports but does not eliminate it.

The proper use of *persistent sockets* should make the need for this change much less likely.

SYMPTOMS:

Client attempts to connect to the TIBCO Patterns Engines fail intermittently. The client may report the Microsoft Windows® socket error:

```
WSAENOBUFS (10055) - Insufficient buffer space
```

SOLUTION:

From `start->run` start the registry editor: `regedit`

> go to key:
>
> `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters`
>
> Add the `DWORD` value: `TcpTimedWaitDelay` if it doesn't exist.
> Set its value to 0x1E (hex) 30 (decimal)
>
> Add the `DWORD` value: `MaxUserPort` if it doesn't exist.
> Set its value to 0x8000 (hex) 32768 (decimal)

Restart your Microsoft Windows® server after making all registry changes.

8.2.2 Turn Off Syn Attack Protection

DESCRIPTION:

On later versions of Microsoft Windows® Denial Of Service (DOS) attack protection is on by default. This feature attempts to detect DOS attacks by looking for and blocking high volumes of connection attempts (specifically it looks at the pending queue of unacknowledged connection attempts). For an application like TIBCO Patterns Engines where every query may generate a new connection heavy client loads can trigger the DOS protection feature causing the OS to forcibly close the connection. Therefore this feature should be turned off for TIBCO Patterns Engines handling high volumes of traffic. (Microsoft makes similar recommendations for its own servers.)

Again the proper use of *persistent sockets* should make the need for this change far less likely.

SYMPTOMS:

Both client and server receive intermittent errors on connections. Client and server may report the Microsoft Windows® socket error:

```
        WSAECONNRESET (10054) - Connection reset by peer.
```

SOLUTION:

From `start->run` start the registry editor: `regedit`

> go to key:
>
> `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters`
>
> Add the `DWORD` value: `SynAttackProtect` if it doesn't exist.
> Set its value to 0

Restart your server after making all registry changes.

8.2.3 Turn Off Scalable Networking Components

DESCRIPTION:

Microsoft's Scalable Networking components can cause problems with connections on systems receiving large numbers of connection requests. It may be necessary to turn off these components.

The proper use of *persistent sockets* should make the need for this change much less likely.

SYMPTOMS:

Both client and server may receive intermittent errors. Client and server may report the Microsoft Windows® socket error:

```
        WSAECONNRESET (10054) - Connection reset by peer.
```

SOLUTION:

If this problem occurs first apply the solution for *Turn Off Syn Attack Protection* as the two problems have identical symptoms and the Syn Attack Protection feature is the more likely cause. If the problem persists disable the Scalable Networking components.

From `start->run` start the registry editor: `regedit`

> go to key:
>
> `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters`
>
> Make sure the `DWORD` value: `EnableTcpChimney` exists and is set to 0.
> Make sure the `DWORD` value: `EnableRSS` exists and is set to 0.
> Make sure the `DWORD` value: `EnableTCPA` exists and is set to 0.

Restart your server after making all registry changes.

9 Running TIBCO Patterns Engines on Unix Variants

To reduce connection overhead, the TIBCO Patterns Engines persists socket connections. It will use up to 1024, or one-third of the file descriptors available to the process, whichever is lower, to persist incoming socket connections. Additionally it will use another one-third to persist outgoing socket connections to other TIBCO Patterns Engines.

Beyond these limits, the server will use transient connections. File descriptor limits should be set (using the `ulimit` command) to at least 1024, with 64K (65536) or higher being preferred.

10 Server Application Interface

The application interface to the TIBCO Patterns Engines is explained in the TIBCO Patterns DevKit manual and the documentation contained in the Java or .Net application interface packages.

A Library Dependencies

Below is a list of shared object or dynamically linked libraries used by the TIBCO Patterns Engines broken down by platform. All of these libraries are standard libraries that should exist on any properly installed and configured system for the given platform. As the list of libraries needed is under the control of the platform vendor this list may change at any time.

AIX 5.3 Power PC - 32 Bit:

Note: the server is built to be backward compatible to 5.1.

libc.a - base "C" library and OS interface.
libcrypt.a - password access and encryption (part of libc).
libpthread.a - POSIX Thread interface.
libpthreads.a - another name for libpthread.a.
libpthreads_compat.a - backward compatibility version of libpthread.a.
libtli.a - supplement to libc.

AIX 5.3 Power PC - 64 Bit:

Note: the server is built to be backward compatible to 5.1.

libc.a - base "C" library and OS interface.
libC.a - processor specific libc routines, and C++ support routines.
libcrypt.a - password access and encryption (part of libc).
libpthread.a - POSIX Thread interface.
libpthreads.a - another name for libpthread.a
libtli.a - supplement to libc.

HP-UX 11.23 Itanium Architecture - 32 Bit:

This should be compatible with most 11 series HP-UX servers running on Itanium based hardware.

libc.so.1 - base "C" library and OS interface.
libdl.so.1 - supports linking to dynamically loaded libraries.
libm.so.1 - basic math functions.
libnsl.so.1 - NIS and related functions.
libpthread.so.1 - POSIX Thread interface.
librt.so.1 - Some timing control functions.
libuca.so.1 - Functions for accessing machine registers and state info.
libunwind.so.1 - Functions for accessing process stack state.
libxti.so.1 - Various HP internal thread and messaging functions.

HP-UX 11.23 Itanium Architecture - 64 Bit:

This should be compatible with most 11 series HP-UX servers running on Itanium machines.

libc.so.1 - base "C" library and OS interface.
libdl.so.1 - supports linking to dynamically loaded libraries.

libm.so.1 - basic math functions.
libnsl.so.1 - NIS and related functions.
libpthread.so.1 - POSIX Thread interface.
librt.so.1 - Some timing control functions.
libuca.so.1 - Functions for accessing machine registers and state info.
libunwind.so.1 - Functions for accessing process stack state.
libxti.so.1 - Various HP internal thread and messaging functions.

Linux - 32 Bit:

This should be compatible with most Intel or AMD x86 compatible hardware platforms running most versions of Linux. Specifically it is built for RedHat Linux.

ld-linux.so.2 - the linux linker/loader.
libc.so.6 - base "C" library and OS interface.
libm.so.6 - basic math functions.
libnsl.so.1 - NIS and related functions.
libpthread.so.0 - POSIX Thread interface.
librt.so.1 - Some timing control functions.

Linux - 64 Bit:

This should be compatible with most Intel or AMD x86 compatible hardware platforms running most versions of Linux. Specifically it is built for RedHat Linux.

ld-linux-x86-64.so.2 - the linux 64 Bit linker/loader.
libc.so.6 (64 Bit) - base "C" library and OS interface.
libm.so.6 (64 Bit) - basic math functions.
libnsl.so.1 (64 Bit) - NIS and related functions.
libpthread.so.0 (64 Bit) - POSIX Thread interface.
librt.so.1 (64 Bit) - Some timing control functions.

Solaris 10 (SunOS 5.10) SPARC - 32 Bit:

libaio.so.1 - Asynchronous I/O routines.
libc_psr.so.1 - Platform specific optimizations of some libc routines.
libc.so.1 - base "C" library and OS interface.
libdoor.so.1 - SUN specific RPC mechanism.
libmd5.so.1 - MD5 Hashing routines.
libmd5_psr.so.1 - Platform specific optimizations of some libmd5 routines.
libmp.so.2 - Multi-precision math library.
libm.so.2 - basic math functions.
libnsl.so.1 - NIS and related functions.
libpthread.so.1 - POSIX Thread interface.
librt.so.1 - Some timing control functions.
libscf.so.1 - Service Configuration Facility library.
libsocket.so.1 - Solaris TCP/IP socket routines.

libuutil.so.1 - some Solaris internal utility functions.

Solaris 10 (SunOS 5.10) SPARC - 64 Bit:

libaio.so.1 (64 Bit) - Asynchronous I/O routines.
libc_psr.so.1 - Platform specific optimizations of some libc routines.
libc.so.1 (64 Bit) - base "C" library and OS interface.
libdoor.so.1 (64 Bit) - SUN specific RPC mechanism.
libmd5.so.1 (64 Bit) - MD5 Hashing routines.
libmd5_psr.so.1 - Platform specific optimizations of some libmd5 routines.
libmp.so.2 (64 Bit) - Multi-precision math library.
libm.so.2 (64 Bit) - basic math functions.
libnsl.so.1 (64 Bit) - NIS and related functions.
libpthread.so.1 (64 Bit) - POSIX Thread interface.
librt.so.1 (64 Bit) - Some timing control functions.
libscf.so.1 (64 Bit) - Service Configuration Facility library.
libsocket.so.1 (64 Bit) - Solaris TCP/IP socket routines.
libuutil.so.1 (64 Bit) - some Solaris internal utility functions.

Microsoft Windows® XP/Server 2003, x86 - 32 Bit:

The libraries listed here are those the TIBCO Patterns Engines calls directly. The functions within these DLLs often link to other Microsoft Windows® DLLs. It is not possible to determine exactly which DLLs will be needed for any particular application. A properly installed Microsoft Windows® OS will always provide the necessary supporting DLLs. Consult Microsoft Corp. for a list of the needed DLLs for your particular machine.

ADVAPI32.dll - basic "C" library routines.
WS2_32.dll - Microsoft Windows® Sockets 2 interface.
KERNEL32.dll - basic Microsoft Windows® Kernel operations.

Microsoft Windows® XP/Server - 64 Bit:

Built for Intel or AMD x86 compatible 64 Bit processors and the Microsoft Windows® XP or Microsoft Windows® Server 2003 64 Bit OS.

The libraries listed here are those the TIBCO Patterns Engines calls directly. The functions within these DLLs often link to other Microsoft Windows® DLLs. It is not possible to determine exactly which DLLs will be needed for any particular application. A properly installed Microsoft Windows® OS will always provide the necessary supporting DLLs. Consult Microsoft Corp. for a list of the needed DLLs for your particular machine.

ADVAPI32.dll - basic "C" library routines.
WS2_32.dll - Microsoft Windows® Sockets 2 interface.
KERNEL32.dll - basic Microsoft Windows® Kernel operations.